TECHNO INTERNATIONAL NEWTOWN

Newtown, Megacity, Kolkata -700156.

Department of Information Technology

Submission for the partial fulfillment of B.Tech Final Year Project- CS881

Project Title

Prepared by

Sahin SK (Roll No:-18700220046)

Atindra Ghosh (Roll No:-18700220073)

Deep Das (Roll No:-18700221126)

Md Aryan (Roll No:-18700220060)

Under The Guidance of

Dr. Ayan Chakraborty

For

Batch:- 2020-2024 Semester : 8 th Year : Jan 2024– June 2024

Stream:- Information Technology Year of Study: 4th

Affiliated to

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WEST BENGAL

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WESTBENGAL

(FORMERLY KNOWN AS WEST BENGAL UNIVERSITY OF TECHNOLOGY)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to **Dr. Ayan Chakraborty**, Principle-In-Charge, Techno India College of Technology, Newtown Megacity, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the cordial support they offered.

Place: Techno India College of Technology, Newtown Megacity

Date: _ _ **03/06/2024** _ _

_____
Sahin SK (Roll No: - 18700220046)

_____
Atindra Ghosh (Roll No: - 18700220073)

_____
Deep Das (Roll No: - 18700221126)

_____
Md Aryan (Roll No: - 18700220060)

Additional Students:
Debsmit Ghosh (18730622050)
Shovan Roy (18730622024)
Santosh Kumar Singh (18700122053)
Souvik Karmakar (18700122047)

Department of Information Technology,
Techno India College of Technology, Newtown Megacity
Kolkata – 700 156
West Bengal, India.

Approval

This is to certify that the project report entitled "**IMMORTAL 2.0**" prepared under my supervision by Sahin Sk. (18700220046), Atindra Ghosh (18700220073), Md Aryan(18700220060), Deep Das(18700221126) be accepted in partial fulfillment for the degree of Bachelor of Technology in Information Technology which is affiliated to Maulana Abul Kalam Azad University of Technology, West Bengal (Formerly known as West Bengal University of Technology).

It is to be understood that by this approval, the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

Project Mentor:

……
Dr. Ayan Chakraborty
Principal-In-Charge
Techno India College of Technology

Additional Mentor:

…..……
Prof. Biswadeep Banerjee
Dept. of Basic Science and Humanities
Techno India College of Technology

External Examiner:

…….

…….
Prof. Sourav Mahapatra,
HOD, Dept. of Information Technology
Techno India College of Technology

Abstract

In this project the critical issue of providing timely and effective medical support in flood-prone rural areas, a significant challenge faced by our government. Our innovative solution involves deploying an advanced medical machine in these regions, capable of collecting vital health parameters such as blood group, body temperature, blood pressure, sugar levels, and SpO2. This data is then transmitted to a centralized server and a dedicated website, accessible to nearby hospitals for remote monitoring and diagnosis by healthcare professionals.

The system is designed to enable swift medical interventions through a streamlined process where doctors can remotely assess patients' conditions and prescribe treatments. Medications are automatically dispatched to the patients via SMS, ensuring they receive timely and clear instructions for their care. The future scope of our project includes expanding the range of monitored parameters, integrating AI and machine learning for predictive analytics, incorporating telemedicine features, and adapting the system for use in various disaster scenarios beyond floods.

Our comprehensive Entity-Relationship (ER) diagram illustrates the system's database architecture, highlighting key entities such as Admin, Doctor, Patient, Appointment, and their interrelationships. The Admin entity manages doctor and patient profiles, while the Appointment entity serves as a central hub for patient-doctor interactions, consolidating all relevant medical documentation.

This project not only exemplifies the use of technology to bridge the healthcare gap in underserved areas but also paves the way for future advancements in emergency medical support. By improving healthcare accessibility and outcomes, particularly in disaster-affected regions, our system aims to make a significant impact on public health. The proposed enhancements and future developments underscore our commitment to leveraging innovative solutions for better healthcare delivery in critical rural areas.

CONTENTS

LIST OF FIGURES

1. Introduction

In our final year project, we're addressing a critical issue faced by our government in handling medical support during floods in rural areas. Our innovative solution involves deploying a machine in flood-prone regions. This machine collects vital medical parameters, including blood group, body temperature, blood pressure, sugar levels, and spo2, offering essential insights for immediate medical attention.

Upon gathering this crucial data, the machine seamlessly updates a centralized server and a dedicated website accessible to nearby hospitals in the flood-affected area. Medical professionals can then review the patient's condition remotely, enabling swift and informed decision-making. The doctor can tailor treatments based on the transmitted parameters, ensuring precise and timely interventions.

Furthermore, our system incorporates a streamlined communication process. Once the treatment is administered, the necessary medications are automatically dispatched to the patient via SMS to their provided phone number. This not only expedites the delivery of essential medicines but also ensures patients receive clear instructions, enhancing the overall efficiency of emergency healthcare in flood-stricken regions.

2. Problem Definition

In response to the critical healthcare challenges faced in remote rural areas, our innovative final year project aims to bridge the gap by introducing an advanced medical support system. We propose the development of a cutting-edge machine strategically placed in critical rural locations. This machine serves as a vital link between individuals in need and healthcare providers.

The machine is designed to gather essential medical parameters such as blood group, body temperature, blood pressure, sugar levels, and spo2. Once collected, this valuable data is promptly transmitted to a centralized server and made accessible via a dedicated website. This real-time information is also shared with the nearest hospital in the flood-prone area, ensuring swift response and efficient medical intervention.

Upon receiving the patient's parameters, medical professionals can remotely assess the situation and prescribe tailored treatments. The prescribed medications are then automatically dispatched to the patient via SMS, including detailed instructions and the necessary medication information. This streamlined process ensures that individuals in critical rural areas receive timely and accurate medical support, overcoming the challenges posed by geographical constraints.

In essence, our business logic revolves around leveraging technology to create a seamless connection between healthcare providers and patients in remote areas. By integrating data collection, transmission, remote diagnosis, and automated medication delivery, we aim to significantly enhance healthcare accessibility and outcomes in critical rural regions, addressing a pressing concern that our government currently grapples with.

3. Architecture:

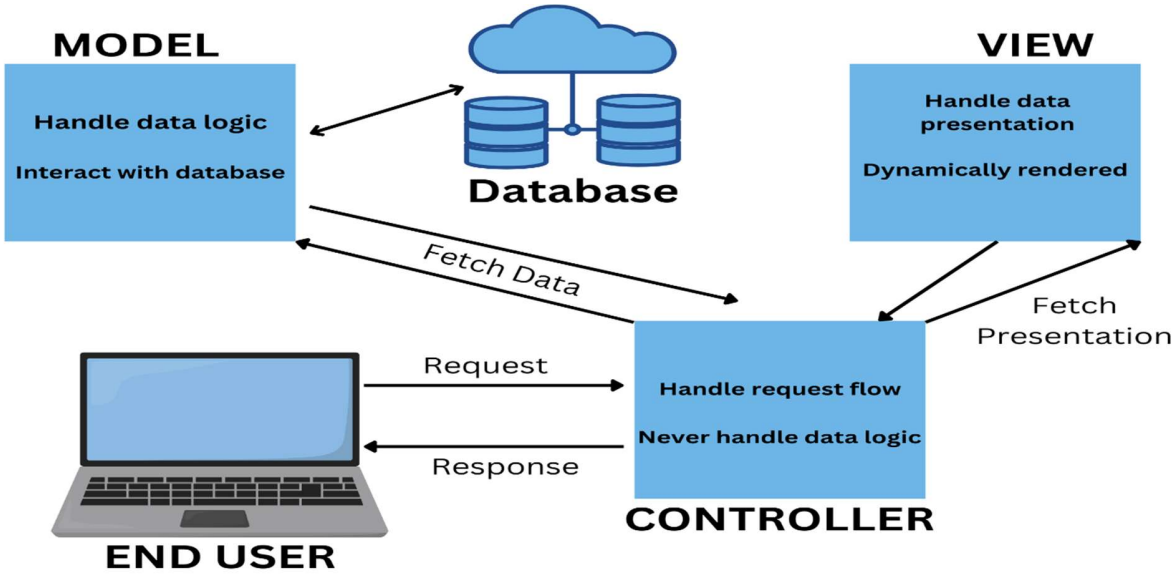

Fig 1: MVC Architecture

4. Data Flow Diagrams(DFD)

4.1 Context Level Diagram

This Level 0 DFD offers a simplified view of how the appointment system interacts with external entities (patient, admin, doctor) and shows the basic flow of information between these entities and the appointment process.

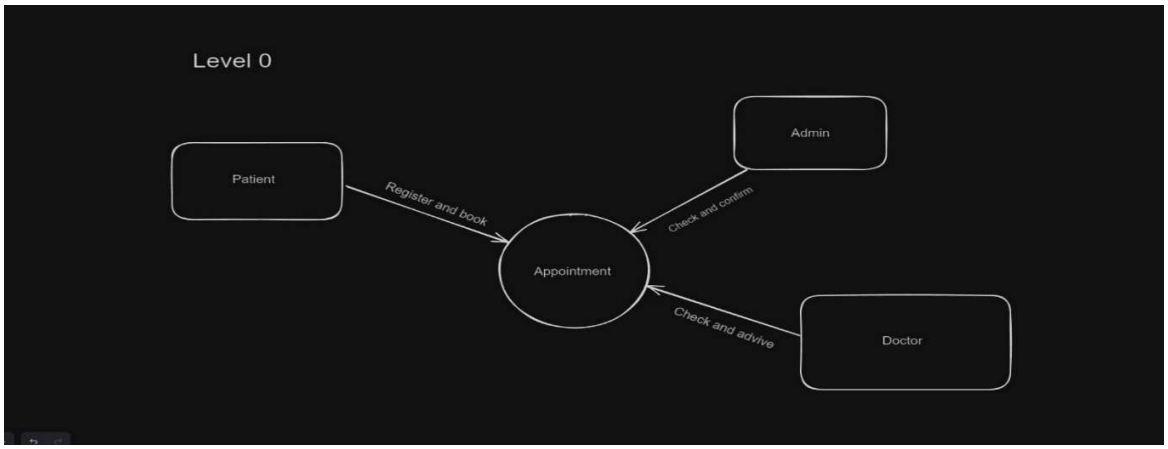

Fig 2: DFD level 0

1. **Entities:**
    - **Patient:** The person who needs to book an appointment.
    - **Admin:** The administrative staff responsible for managing and confirming appointments.
    - **Doctor:** The healthcare provider who checks and advises patients during their appointments.
2. **Processes and Data Stores:**
    - **Appointment:** This central process or data store handles the core functionality of booking and managing appointments. It interacts with the patient, admin, and doctor.

3. **Interactions:**
    - **Patient to Appointment: "Register and book"**
        - The patient initiates the process by registering and booking an appointment. This involves providing necessary details like personal information, preferred date and time, and any other relevant information needed for the appointment.
    - **Admin to Appointment: "Check and confirm"**
        - The admin checks the details of the appointment to confirm its validity. This could involve verifying the patient's information, ensuring the chosen time slot is available, and finalizing the booking in the system.
    - **Doctor to Appointment: "Check and advise"**
        - The doctor accesses the appointment details to prepare for the patient's visit. During the appointment, the doctor will check the patient's condition and provide medical advice or treatment.

Overall Flow:

- The process starts with the patient registering and booking an appointment.
- The admin then checks the appointment details and confirms the booking.
- Finally, the doctor checks the confirmed appointment details to prepare for the patient's visit and provide the necessary medical advice or treatment during the appointment.

4.2 First Level DFD:

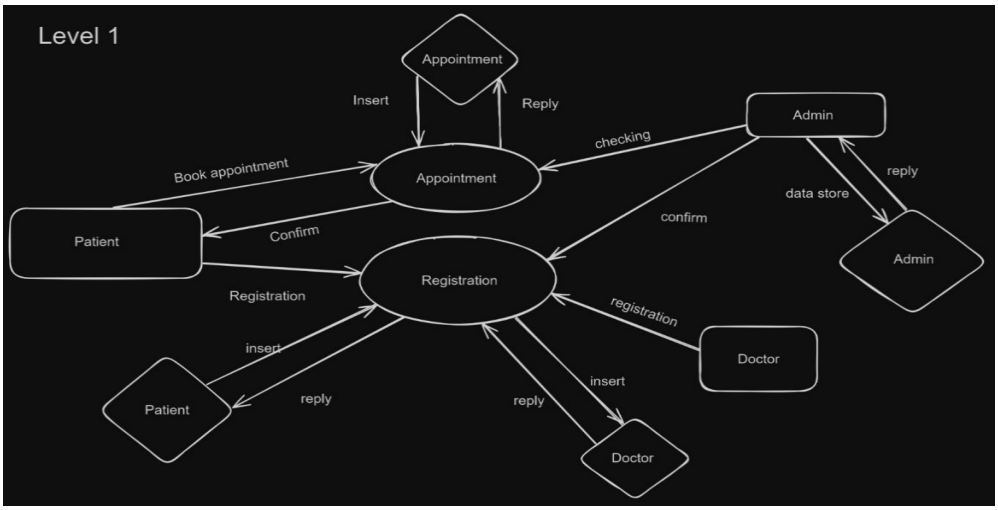

Fig 3: DFD level 1

External Entities:

Patient: The patient is an external entity that interacts with the system. They initiate the process of booking an appointment, provide personal and medical information, and receive confirmations.

Admin: The administrative staff is another external entity that interacts with the system. They manage the appointment system, check appointment details, and update patient information.

Doctor: The doctor is an external entity that interacts with the system. They register patient information, manage appointments, and receive confirmations.

Processes:

Book Appointment (Process 1): This process represents the patient booking an appointment. It involves the following steps:

The patient initiates the booking process by providing their personal and medical information.

The system checks the availability of the doctor and the appointment schedule.
The system confirms the appointment details with the patient.
Registration (Process 2): This process represents the registration of a new patient or the updating of existing patient information. It involves the following steps:

The patient provides their personal and medical information.

The system stores the patient information in the data store.
The system updates the patient's information in the data store.
Appointment (Process 3): This process represents the management of appointments, including scheduling, rescheduling, and cancellations. It involves the following steps:

The system schedules a new appointment or reschedules an existing one.
The system cancels an appointment if necessary.
The system updates the appointment schedule in the data store.
Data Stores:

Data Store: The data store is a central repository that stores patient information, appointment details, and other system data. It provides a single source of truth for the system and ensures data integrity.
Data Flows:

Book Appointment → Patient: The patient initiates the booking process by providing their personal and medical information.
Patient → Book Appointment: The patient confirms the appointment details.
Book Appointment → Registration: The patient's information is collected and stored in the data store.
Registration → Insert: The patient's details are added to the system.
Admin → Check: The administrative staff checks the appointment details.
Check → Reply: The system sends a confirmation to the patient.
Registration → Doctor: The doctor registers a patient's information.
Doctor → Insert: The appointment details are added to the system.
Insert → Reply: The system sends a confirmation to the doctor.
Data Store → Admin: The administrative staff retrieves patient information and appointment details from the data store.
Data Store → Doctor: The doctor retrieves patient information and appointment details from the data store.

Second Level DFD

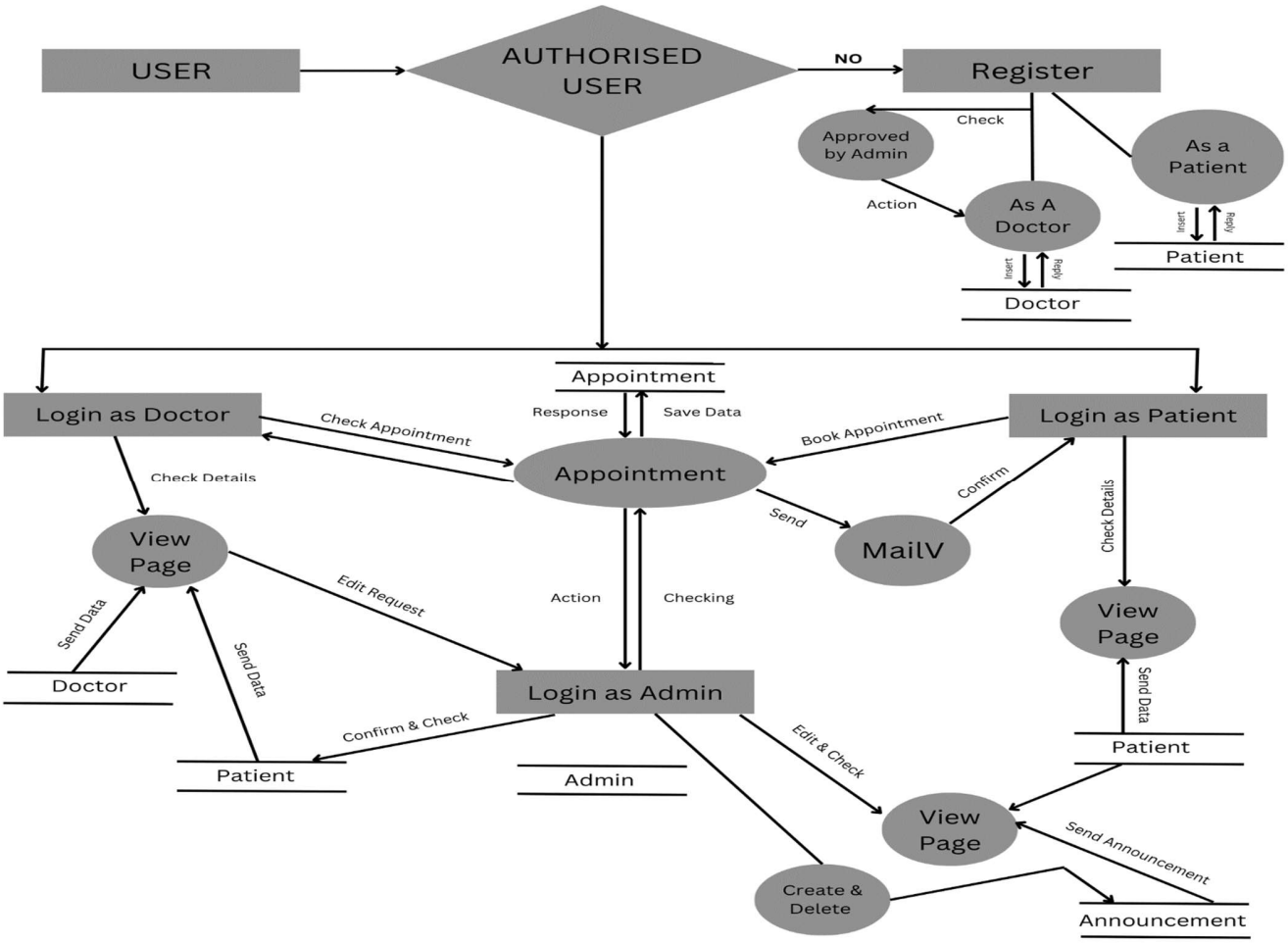

Fig 4: DFD level 3

4.3 Second level DFD:

Overview

- The Level 2 DFD provides a detailed view of the data processes and data flows within the healthcare management system.
- It breaks down the high-level processes into more specific sub-processes, showing how data moves between these processes, data stores, and external entities.

Entities and Processes

1. **Admin**
    - **Attributes**: Admin_ID, Name, Email_Id, Password, Gov.ID, Address, Mob_No

- **Manage Process**: The admin manages the overall system, including approvals, announcements, and user management.
- **Approval Process**: The admin handles approval of doctors and other critical tasks.
- **Data Flow**:
  - Receives data from Admin to manage processes.
  - Sends and receives data from the Announcement process.
  - Sends and receives data from the Approval process.

2. **Doctor**
   - **Attributes**: Author.No, Gov.ID, Mobile.No, Password, Email_Id, Address, Doc_Name
   - **Data Flow**:
     - Interacts with Admin for approvals and management.
     - Involved in the Appointment process for booking and managing appointments.
3. **Patient**
   - **Attributes**: Patient_ID, Name, Address, Mobile.No, Email_Id, Password, Adher.No, Age
   - **Booking Process**: Patients book appointments with doctors.
   - **Check Process**: Involves checking patient health parameters and appointment details.
   - **Data Flow**:
     - Sends and receives data from the Appointment process.
     - Sends and receives data from the Health Parameter process.

Processes and Data Stores

1. **Announcement Process**
   - Manages announcements for patients and doctors.
   - **Data Flow**:
     - Receives and sends announcements to Admin.
     - Sends announcements to patients and doctors.
2. **Approval Process**
   - Admin approves doctors and critical tasks.
   - **Data Flow**:
     - Receives data from Admin.
     - Sends approvals to the Doctor process.
3. **Appointment Process**
   - Manages doctor appointments5.
   - **Attributes**: Appoint-D/T, Time, Doctor_ID, Appointment_ID
   - **Data Flow**:
     - Receives appointment details from Booking.
     - Sends appointment details to doctors.
     - Sends and receives data from the Check process.
4. **Document Process**
   - Manages document storage and retrieval for doctors and patients.

- **Attributes**: Author.No, Gov.ID, Mobile.No, Password, Email_Id, Address, Doc_Name
- **Data Flow**:
  - Stores and retrieves documents related to appointments and patient health records.

5. **Check Process**
   - Involves verifying patient details and health parameters during appointments.
   - **Attributes**: Appoint-D/T, Time, Doctor_ID, Appointment_ID
   - **Data Flow**:
     - Receives data from the Appointment process.
     - Sends and receives data to/from the Health Parameter process.
6. **Health Parameter Process**
   - Manages patient health details such as Blood_GRP, Height, Weight, Blood_Prss, Blood_Sugar, Spo2.
   - **Data Flow**:
     - Receives data from the Check process.
     - Sends health parameters to the Patient process.

Data Stores

1. **Admin Data Store**
   - Stores admin-related information like Admin_ID, Name, Email_Id, Password, Gov.ID, Address, Mob_No.
2. **Doctor Data Store**
   - Stores doctor-related information like Author.No, Gov.ID, Mobile.No, Password, Email_Id, Address, Doc_Name.
3. **Patient Data Store**
   - Stores patient-related information like Patient_ID, Name, Address, Mobile.No, Email_Id, Password, Adher.No, Age.
4. **Appointment Data Store**
   - Stores appointment-related information like Appoint-D/T, Time, Doctor_ID, Appointment_ID.
5. **Document Data Store**
   - Stores documents related to doctors and patients.
6. **Health Parameter Data Store**
   - Stores health parameters of patients like Blood_GRP, Height, Weight, Blood_Prss, Blood_Sugar, Spo2.

5. Entity Relationship Diagrams

In this section you need to specify the entity –relationship diagram. You need to identify & specify all the entities involved and then with a help of a diagram represent the same. Please ensure in the diagram the following properties are mentioned with proper notation: -

a) Primary key
b) Weak entity(if any)
c) Derived attribute(if any)
d) Cardinality of the relationships.

Entity-Relationship (ER) Diagram Explanation

The ER diagram represents the structure and relationships within our advanced medical support system, designed for use in flood-prone rural areas. This diagram outlines the key entities, their attributes, and the interactions between them, providing a comprehensive overview of the system's database architecture.

Entities and Their Attributes:

1. **Admin**
   - **Attributes**:
     - Admin_ID: Unique identifier for the admin.
     - Name: Name of the admin.
     - Email_ID: Email address of the admin.
     - Password: Password for admin access.
     - Address: Address of the admin.
     - Mob_No: Mobile number of the admin.
     - Gov_ID: Government-issued ID of the admin.
2. **Doctor**
   - **Attributes**:
     - Doctor_ID: Unique identifier for the doctor.
     - Doc_Name: Name of the doctor.
     - Email_ID: Email address of the doctor.
     - Password: Password for doctor access.
     - Address: Address of the doctor.
     - Mobile_No: Mobile number of the doctor.
     - Gov_ID: Government-issued ID of the doctor.
     - Adhar_No: Aadhaar number (unique identifier in India).
3. **Patient**
   - **Attributes**:
     - Patient_ID: Unique identifier for the patient.
     - Name: Name of the patient.
     - Email_ID: Email address of the patient.
     - Password: Password for patient access.

        - Address: Address of the patient.
        - Mobile_No: Mobile number of the patient.
        - Adher_No: Aadhaar number (unique identifier in India).
        - Age: Age of the patient.
        - Health_Parameter: Collection of vital health parameters.
            - Blood_GRP: Blood group of the patient.
            - Height: Height of the patient.
            - Weight: Weight of the patient.
            - Blood_Prss: Blood pressure of the patient.
            - Blood_Sugar: Blood sugar levels of the patient.
            - Spo2: Oxygen saturation level of the patient
4. **Appointment**
    - **Attributes**:
        - Appointment_ID: Unique identifier for the appointment.
        - Patient_ID: Identifier linking to the patient.
        - Doctor_ID: Identifier linking to the doctor.
        - Time: Scheduled time of the appointment.
        - Appoin_D/T: Date and time of the appointment.
        - Document: Collection of documents related to the appointment.
            - Prescription: Prescriptions issued during the appointment.
            - Voice: Voice notes or messages.
            - Report: Medical reports and findings.
5. **Announcement**
    - **Attributes**:
        - Patient_Anno: Announcements directed towards patients.
        - Doctor_Anno: Announcements directed towards doctors.

Relationships:

1. **Admin to Doctor**:
    - **Relationship Type**: Manage
    - **Description**: Admin manages doctor profiles and their approvals. Each admin can manage multiple doctors, ensuring that doctors are authorized and their details are up-to-date.
2. **Admin to Patient**:
    - **Relationship Type**: Manage
    - **Description**: Admin manages patient profiles. This includes the verification and maintenance of patient data, ensuring all necessary health parameters are recorded.
3. **Admin to Announcement**:
    - **Relationship Type**: Manage
    - **Description**: Admin creates and manages announcements for both patients and doctors. This ensures important information is disseminated efficiently.
4. **Doctor to Appointment**:
    - **Relationship Type**: Approval

- **Description**: Doctors approve appointments. This involves reviewing the patient's medical parameters and scheduling appropriate times for consultations.

5. **Patient to Appointment**:
   - **Relationship Type**: Booking
   - **Description**: Patients book appointments with doctors. They select available time slots and submit their health parameters for review.
6. **Appointment to Document**:
   - **Relationship Type**: Contains
   - **Description**: Each appointment can contain multiple documents, including prescriptions, voice notes, and medical reports. This ensures all relevant medical information is consolidated and accessible.
7. **Patient to Health_Parameter**:
   - **Relationship Type**: Check
   - **Description**: Patients provide their health parameters, which are regularly checked and updated. This data is crucial for remote diagnosis and treatment.

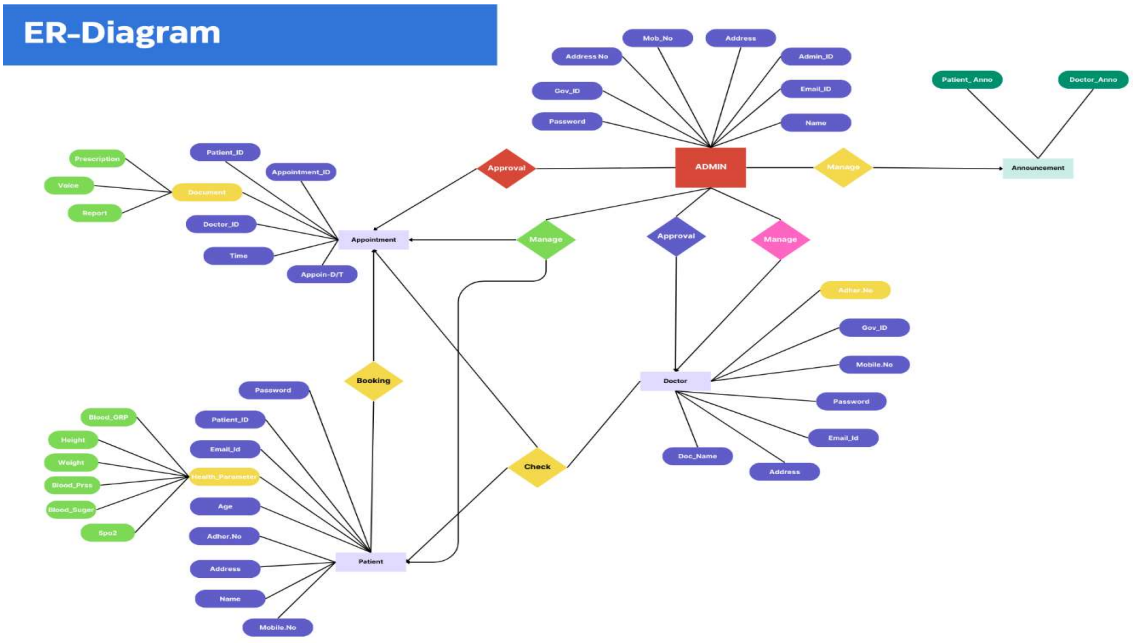

Fig 5: ER Diagram

Explanation of the ER Diagram:

The ER diagram visually captures the interconnectedness of the system's entities and their attributes, highlighting the workflow and data management processes. Each entity represents a crucial component of the system, and their relationships ensure seamless interaction between patients, doctors, and administrators.

1. **Central Role of Admin**:
    - The Admin entity plays a pivotal role, managing both doctors and patients. Admins oversee the approval processes, maintain the integrity of user data, and ensure that all system operations are functioning smoothly. They also manage announcements, keeping all stakeholders informed.
2. **Patient and Doctor Interactions**:
    - Patients interact with the system primarily through booking appointments and providing health parameters. Doctors, on the other hand, are responsible for reviewing and approving these appointments, leveraging the health data provided to make informed medical decisions.
3. **Appointments as a Central Hub**:
    - The Appointment entity acts as a central hub, linking patients to doctors and consolidating all relevant medical documentation. This entity ensures that all interactions are recorded and accessible for future reference.
4. **Health Data Management**:
    - The system places significant emphasis on the collection and management of health parameters. Accurate and up-to-date health data is essential for remote diagnosis and treatment, making the Health_Parameter attributes crucial for the system's efficacy.
5. **Seamless Communication through Announcements**:
    - Announcements serve as a communication bridge, ensuring that patients and doctors receive timely and relevant information. This feature enhances the system's responsiveness and user engagement.

Overall, this ER diagram encapsulates the core functionalities and data flow within our advanced medical support system, illustrating how technology can be leveraged to enhance healthcare accessibility and efficiency in disaster-prone rural areas.

6. Use Case Diagram:

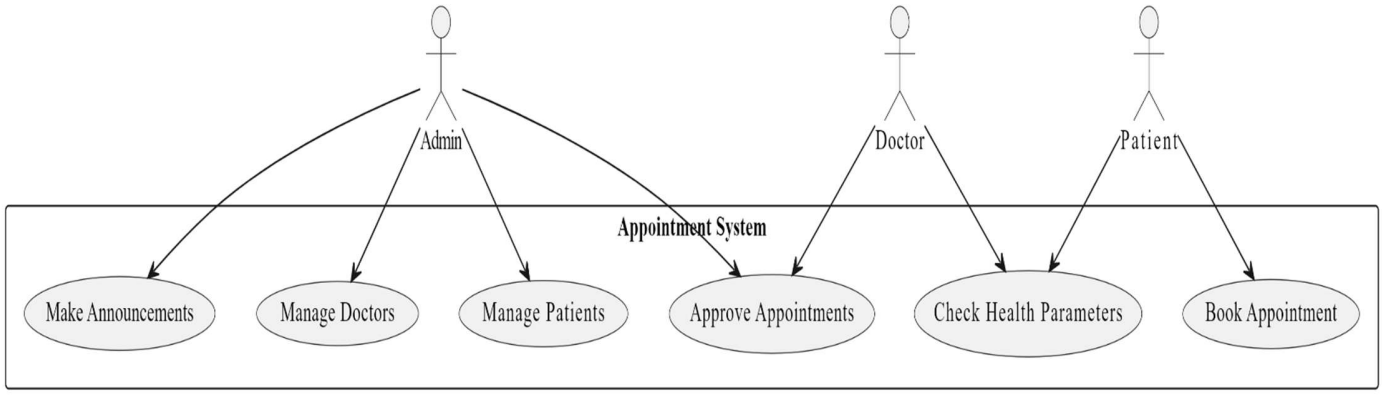

Fig 6: Use Case Diagram

Actors:

1. **Admin**

- Represents the administrative user who has elevated permissions to manage various aspects of the system.

2. **Doctor**
   - Represents the medical professional who interacts with the system to approve appointments and check health parameters of patients.
3. **Patient**
   - Represents the end-user (patient) who uses the system to book appointments and check their health parameters.

Use Cases:

1. **Make Announcements**
   - **Actor:** Admin
   - **Description:** This use case allows the admin to make announcements within the system. These announcements can be directed towards doctors, patients, or both.
2. **Manage Doctors**
   - **Actor:** Admin
   - **Description:** This use case involves the admin managing doctor-related data and permissions. This can include adding new doctors, updating their information, or removing them from the system.
3. **Manage Patients**
   - **Actor:** Admin
   - **Description:** This use case involves the admin managing patient-related data. This includes adding new patients, updating their details, or managing their records.
4. **Approve Appointments**
   - **Actor:** Doctor
   - **Description:** In this use case, doctors review and approve appointment requests made by patients. This ensures that the doctor confirms availability and suitability for the requested time slots.
5. **Check Health Parameters**
   - **Actor:** Doctor, Patient
   - **Description:** This use case allows both doctors and patients to access and review health parameters. For doctors, it involves checking the patients' health metrics for monitoring and diagnosis. For patients, it involves viewing their own health data for personal tracking.
6. **Book Appointment**
   - **Actor:** Patient
   - **Description:** This use case enables patients to book appointments with doctors. The patient selects a preferred time slot and the specific doctor they wish to consult.

Relationships:

- **Admin**:
  - **Manages Announcements**: The admin creates and manages announcements that can be viewed by doctors and patients.
  - **Manages Doctors**: The admin oversees doctor-related data management.

- **Manages Patients**: The admin oversees patient-related data management.

- **Doctor**:
  - **Approves Appointments**: Doctors have the capability to approve or reject appointment requests made by patients.
  - **Checks Health Parameters**: Doctors review health parameters of patients to monitor and provide medical care.
- **Patient**:
  - **Checks Health Parameters**: Patients can view their own health metrics.
  - **Books Appointments**: Patients can schedule appointments with doctors for consultations.

Key Points:

- **Actors**:
  - Each actor has specific roles and permissions within the system.
  - The **Admin** has the highest level of access, with capabilities to manage both doctors and patients.
  - The **Doctor** has access primarily to appointment approvals and health parameter monitoring.
  - The **Patient** interacts with the system mainly to book appointments and check their own health data.
- **Use Cases**:
  - Use cases define the interactions between the actors and the system.
  - They describe the primary functions of the system from the user's perspective.
- **Relationships**:
  - Arrows indicate the direction of interaction between actors and use cases.
  - They show which actor initiates or performs which use case.

7. Class Diagram(Optional)

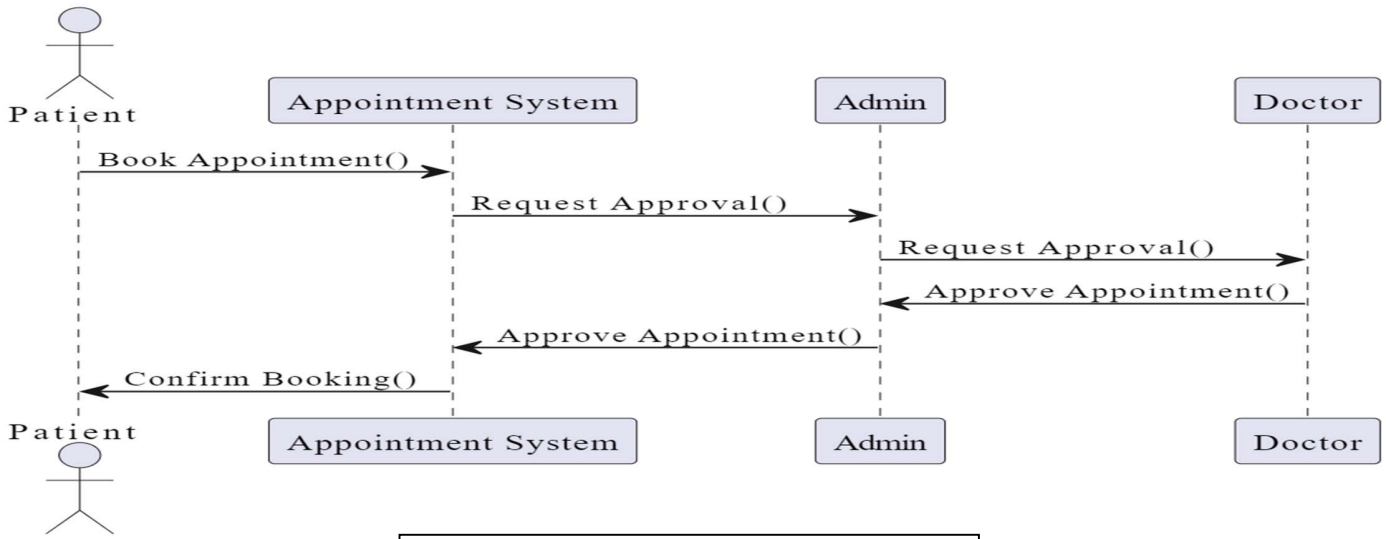

Fig 7: Class Diagram

Actors and Objects:

1. **Patient**:
   - The user who initiates the booking of an appointment.
2. **Appointment System**:
   - The system that handles the booking process and coordinates between the patient, admin, and doctor.
3. **Admin**:
   - The administrative user who approves the appointment requests before they are sent to the doctor.
4. **Doctor**:
   - The medical professional who gives the final approval for the appointment.

Sequence of Interactions:

1. **Book Appointment()**:
   - **Initiator**: Patient
   - **Description**: The patient sends a request to the Appointment System to book an appointment.
2. **Request Approval()**:
   - **Initiator**: Appointment System
   - **Recipient**: Admin
   - **Description**: The Appointment System forwards the booking request to the Admin for approval.
3. **Request Approval()**:
   - **Initiator**: Admin
   - **Recipient**: Doctor
   - **Description**: The Admin forwards the booking request to the Doctor for final approval. This step ensures that the doctor is informed and can confirm their availability.
4. **Approve Appointment()**:
   - **Initiator**: Doctor
   - **Recipient**: Admin
   - **Description**: The Doctor approves the appointment request. This indicates that the doctor has confirmed their availability for the requested time slot.
5. **Approve Appointment()**:
   - **Initiator**: Admin
   - **Recipient**: Appointment System
   - **Description**: The Admin receives the doctor's approval and updates the Appointment System to reflect the approval.
6. **Confirm Booking()**:
   - **Initiator**: Appointment System
   - **Recipient**: Patient

- **Description**: The Appointment System confirms the booking to the patient, indicating that the appointment has been successfully approved and scheduled.

Key Points:

- **Initiation**: The process begins with the patient booking an appointment.
- **Approval Flow**: The appointment request follows a hierarchical approval flow:
  - The request is first sent to the Admin.
  - The Admin forwards it to the Doctor.
  - The Doctor's approval is sent back through the Admin to the Appointment System.
- **Confirmation**: Once the appointment is approved by the doctor, the system confirms the booking to the patient.
- **Coordination**: The Appointment System acts as a coordinator, handling requests and updates between the patient, admin, and doctor.
- **Role of Admin**: The admin acts as an intermediary, ensuring that requests are properly managed and forwarded to the doctor for final approval.
- **Final Authority**: The final authority to approve or reject an appointment lies with the doctor, ensuring that their schedule and availability are respected.

Detailed Interaction Flow:

- **Patient Interaction**: The patient interacts only with the Appointment System, making the initial request and receiving the final confirmation.
- **Appointment System's Role**: The Appointment System manages the flow of information, ensuring that requests and approvals are properly routed.
- **Admin's Role**: The Admin is responsible for preliminary approval and ensuring that requests are forwarded to the appropriate doctor.
- **Doctor's Role**: The doctor provides the final approval, confirming their availability and willingness to take the appointment.

8. Sequence Diagram:

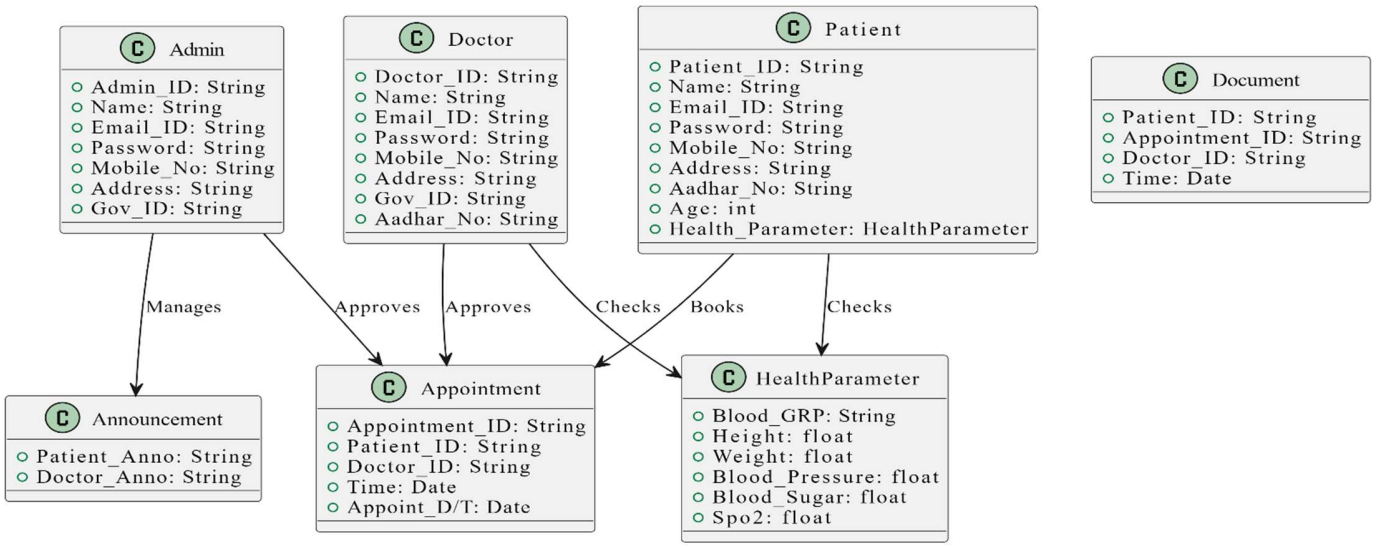

Fig 8: Sequence Diagram

Entities and Attributes:

1. **Admin**
   - **Attributes:**
     - Admin_ID: String
     - Name: String
     - Email_ID: String
     - Password: String
     - Mobile_No: String
     - Address: String
     - Gov_ID: String
2. **Doctor**
   - **Attributes:**
     - Doctor_ID: String
     - Name: String
     - Email_ID: String
     - Password: String
     - Mobile_No: String
     - Address: String
     - Gov_ID: String
     - Aadhar_No: String
3. **Patient**
   - **Attributes:**
     - Patient_ID: String
     - Name: String
     - Email_ID: String
     - Password: String

        - Mobile_No: String
        - Address: String
        - Aadhar_No: String
        - Age: int
        - Health_Parameter: HealthParameter
4. **Appointment**
    - **Attributes:**
        - Appointment_ID: String
        - Patient_ID: String
        - Doctor_ID: String
        - Time: Date
        - Appoint_D/T: Date
5. **HealthParameter**
    - **Attributes:**
        - Blood_GRP: String
        - Height: float
        - Weight: float
        - Blood_Pressure: float
        - Blood_Sugar: float
        - Spo2: float
6. **Document**
    - **Attributes:**
        - Patient_ID: String
        - Appointment_ID: String
        - Doctor_ID: String
        - Time: Date
7. **Announcement**
    - **Attributes:**
        - Patient_Anno: String
        - Doctor_Anno: String

Relationships:

1. **Admin Manages Announcement**
    - The Admin entity manages the Announcement entity. This indicates that admins have the authority to manage announcements for both patients and doctors.
2. **Doctor Approves Appointment**
    - The Doctor entity approves the Appointment entity. This implies that doctors have the capability to approve appointments made by patients.
3. **Patient Books Appointment**
    - The Patient entity books an Appointment. This shows the action where patients schedule appointments with doctors.
4. **Doctor Checks HealthParameter**
    - The Doctor entity checks the HealthParameter entity. This relationship indicates that doctors monitor or record health parameters of patients.
5. **Patient Checks HealthParameter**
    - The Patient entity checks the HealthParameter entity, indicating that patients can view or track their own health parameters.
6. **Document Records Appointment Details**

- The Document entity is used to record details related to the Patient, Appointment, and Doctor. This ensures that records of appointments are maintained.

Key Points:

- **Admin** has a management role over **Announcements**.
- **Doctors** play a key role in approving appointments and monitoring health parameters.
- **Patients** are central to the system, booking appointments, and tracking their health parameters.
- **Health Parameter** entity contains various health metrics crucial for patient health monitoring.
- **Appointments** connect patients and doctors, ensuring a scheduled interaction for consultations.
- **Documents** ensure that all interactions and health-related records are maintained systematically.

9. .HARDWARE:

9.1 Block Diagram:

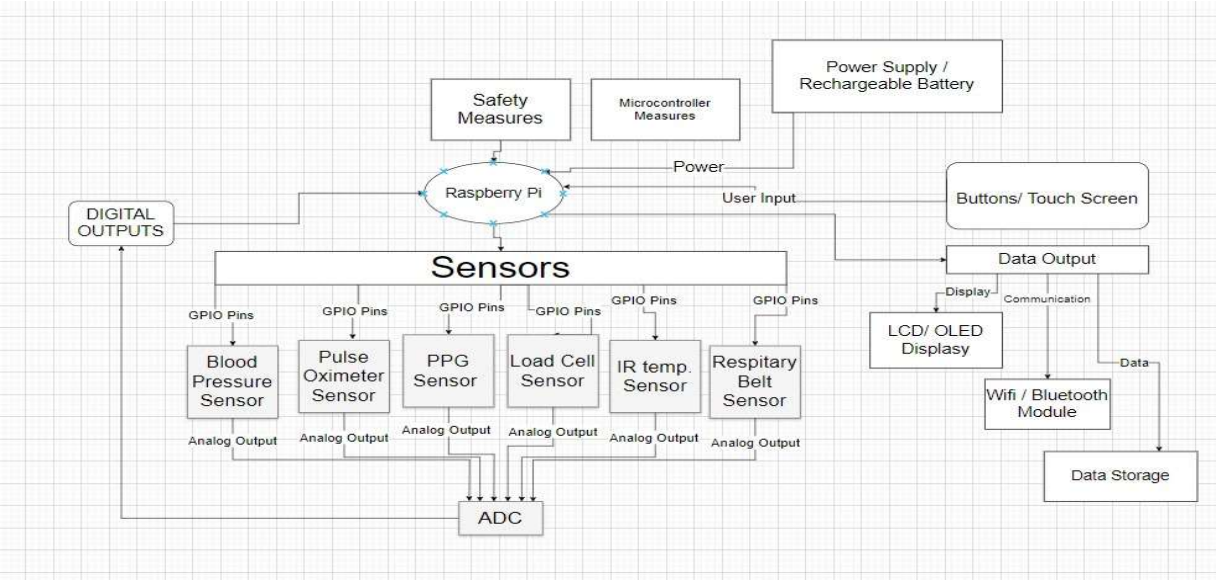

Fig 9: Block diagram (1)

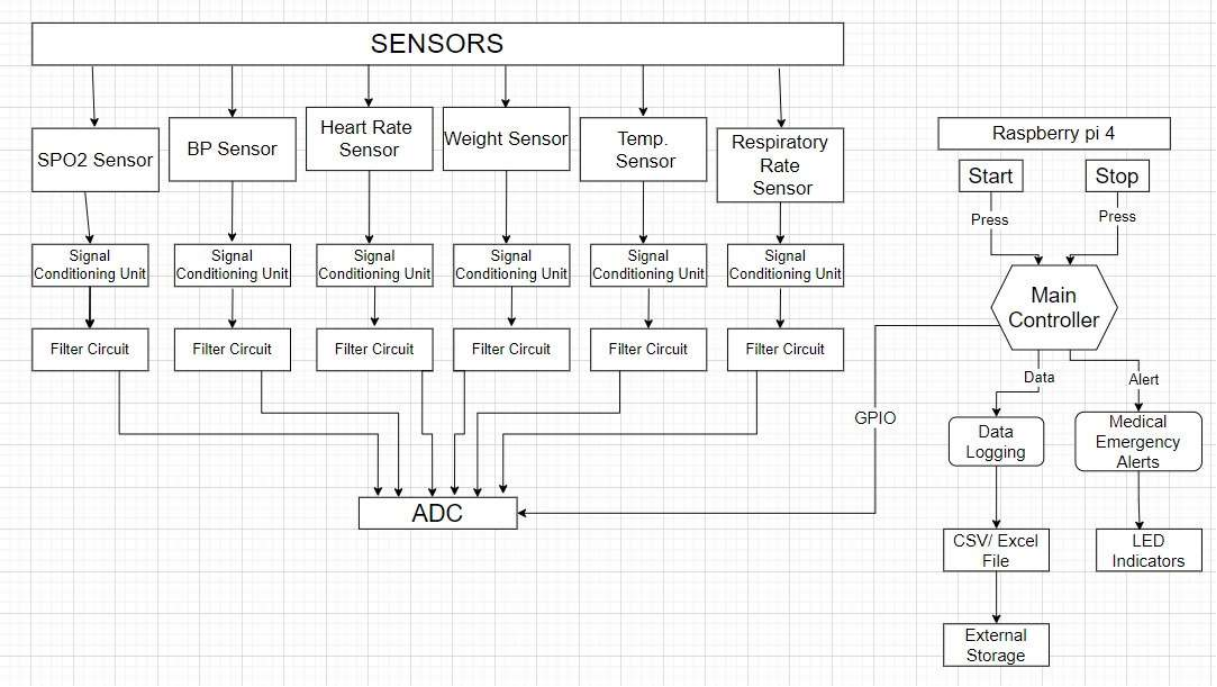

Fig 10: Block diagram (2)

9.2. Hardware Description:

Sensors: There are 6 sensors used in the device.

1. Pulse oximeter sensor.

2. Blood pressure sensor

3. (PPG) Sensor for heart rate

4. (FSR) Force-sensitive resistor for weight

5. Infrared Temperature Sensor for temperature

6. Respiratory Belt for respiratory rate

Circuit:

1. Power supply: Implement a power supply subsystem suitable for a portable device, considering rechargeable batteries or power adapter options.

2. Microcontroller/Processor: Utilize a Raspberry Pi as the main processing unit with GPIO pins for interfacing with various sensors.

10. Future Scope of the Project

1. Expanded Medical Parameter Monitoring

Description: Future iterations of our system can include sensors and diagnostic tools for a broader range of medical parameters. This can involve integrating:

- **ECG (Electrocardiography):** Monitoring heart activity to detect arrhythmias or other cardiac issues.
- **Respiratory Rate:** Tracking breathing patterns to identify respiratory distress.
- **Real-time Imaging (e.g., Ultrasound):** Enabling remote imaging to assist in diagnosing internal injuries or conditions.

Impact: These enhancements would provide healthcare professionals with a more comprehensive understanding of the patient's condition, allowing for more precise and informed medical interventions, which is crucial in emergency scenarios.

2. AI and Machine Learning Integration

Description: Incorporating AI and machine learning can significantly enhance the system's predictive and diagnostic capabilities. Potential applications include:

- **Predictive Analytics:** Using historical data to predict potential health crises before they occur.
- **Pattern Recognition:** Identifying subtle changes in health parameters that could indicate emerging issues.
- **Personalized Treatment Plans:** Tailoring interventions based on individual health data and trends.

Impact: These technologies would allow for proactive healthcare management, reducing the risk of severe medical emergencies and improving overall patient outcomes through timely and personalized interventions.

3. Telemedicine Integration

Description: Adding telemedicine features would enable live video consultations between patients and healthcare providers. This would include:

- **Real-time Video Calls:** Allowing doctors to visually assess patients and provide immediate advice.
- **Interactive Communication:** Enabling two-way communication for better diagnosis and reassurance.

Impact: Telemedicine integration would bridge the gap between remote patients and healthcare providers, offering immediate support and potentially lifesaving advice, thereby enhancing the quality and timeliness of medical care.

4. Scalability to Other Disaster-Prone Areas

Description: The system can be adapted for use in various disaster scenarios beyond floods, such as:

- **Earthquakes:** Deploying in areas with a high risk of seismic activity.
- **Cyclones:** Providing medical support in regions frequently affected by Cyclone.
- **Wildfires:** Assisting communities impacted by large-scale fires.

Impact: This scalability ensures that the system can be a versatile tool for emergency medical support, providing crucial healthcare services across different types of disaster-stricken areas, thus broadening its utility and impact.

5. Enhanced Communication Networks

Description: Future versions of the system could leverage advanced communication technologies to improve data transmission:

- **5G Networks:** Providing faster and more reliable connectivity.
- **Mesh Networks:** Ensuring network resilience and coverage in remote or infrastructure-damaged areas.

Impact: Improved communication networks would enhance the system's efficiency, ensuring that medical data is transmitted quickly and reliably, even in challenging conditions, thereby improving response times and coordination.

6. Mobile Application Development

Description: Developing a dedicated mobile app would facilitate better user interaction and accessibility:

- **Patient Interface:** Allowing patients to receive real-time updates, instructions, and health status alerts.
- **Doctor Interface:** Enabling healthcare providers to monitor patient data and communicate more efficiently.

Impact: A mobile app would provide a user-friendly platform for both patients and doctors, enhancing the overall user experience and ensuring better engagement with the system.

7. Integration with Government and Health Systems

Description: Collaborating with government health systems and agencies can lead to a more integrated emergency response:

- **National Health Databases:** Connecting with existing health records for better-informed medical decisions.
- **Emergency Services Coordination:** Streamlining operations with emergency responders for resource allocation.

Impact: This integration would create a more cohesive and efficient healthcare response system, leveraging existing infrastructures to maximize the system's effectiveness and reach.

8. Community Training and Engagement

Description: Implementing training programs for local communities would ensure better usage and engagement:

- **Educational Workshops:** Teaching residents how to use the machines effectively.
- **Local Health Volunteers:** Training local volunteers to assist in operating and maintaining the machines.

Impact: Educated and engaged communities would contribute to the accuracy and efficacy of the system, ensuring that medical support is optimally utilized and maintained.

9. Data Security and Privacy Enhancements

Description: As the system deals with sensitive medical information, enhancing data security is crucial:

- **Advanced Encryption:** Ensuring data is protected during transmission and storage.
- **Secure Access Protocols:** Implementing multi-factor authentication and other security measures.

Impact: Strengthening data security would protect patient privacy and build trust in the system, ensuring that sensitive information is safeguarded against unauthorized access.

10. Sustainability and Self-Powered Systems

Description: To ensure continuous operation during power outages, integrating sustainable energy solutions is essential:

- **Solar Panels:** Providing a renewable energy source.
- **Hand-Crank Generators:** Offering a manual power option in emergencies.

Impact: Sustainable energy solutions would enhance the system's resilience, ensuring it remains operational during power failures, which are common in disaster scenarios, thereby maintaining continuous medical support.

By exploring and implementing these future advancements, your project can significantly evolve, providing more comprehensive, reliable, and accessible healthcare support to rural and disaster-affected regions, thereby addressing critical healthcare challenges effectively.

- ## Conclusion

In conclusion, our final year project presents a transformative solution to the critical issue of providing medical support in flood-prone rural areas. By deploying a sophisticated machine that collects vital medical parameters and seamlessly integrates with a centralized server, we bridge the gap between remote patients and healthcare providers. This system not only ensures swift and informed medical responses but also automates the dispatch of necessary medications, enhancing the efficiency and effectiveness of emergency healthcare.

Looking forward, the future scope of our project is vast and promising. By expanding the range of monitored medical parameters, integrating AI and machine learning, incorporating telemedicine, and adapting the system for other disaster scenarios, we can significantly enhance its capabilities and impact. Improved communication networks, a dedicated mobile application, and robust integration with government health systems will further streamline operations and extend the system's reach. Community training and enhanced data security measures will ensure the system's reliability and user trust, while sustainable power solutions will guarantee continuous operation even in challenging conditions.

Our vision is to leverage advanced technology to create a resilient, comprehensive, and accessible healthcare support system. By addressing the pressing healthcare challenges in underserved and disaster-affected regions, we aim to improve healthcare accessibility and outcomes, ultimately contributing to better health and well-being for vulnerable populations. This project not only exemplifies innovative problem-solving but also paves the way for future advancements in emergency medical support, reaffirming our commitment to making a meaningful impact in the field of healthcare.

- **Bibliography**

MDN Web Docs: "HTML: HyperText Markup Language" https://developer.mozilla.org/en-US/docs/Web/HTML
MDN Web Docs: "CSS: Cascading Style Sheets" https://developer.mozilla.org/en-US/docs/Web/CSS
MDN Web Docs: "JavaScript" https://developer.mozilla.org/en-US/docs/Web/JavaScript
W3Schools: "HTML Tutorial" https://www.w3schools.com/html/
W3Schools: "CSS Tutorial" https://www.w3schools.com/css/
W3Schools: "JavaScript Tutorial" https://www.w3schools.com/js/
Duckett, J. (2014). HTML and CSS: Design and Build Websites. Wiley.
Duckett, J. (2014). JavaScript and JQuery: Interactive Front-End Web Development. Wiley.
Bootstrap Official Documentation: "Introduction" https://getbootstrap.com/docs/5.1/getting-started/introduction/
W3Schools: "Bootstrap Tutorial" https://www.w3schools.com/bootstrap/
Spurlock, J. (2013). Bootstrap: Responsive Web Development. O'Reilly Media.
Python Official Documentation: "Python 3.x Documentation" https://docs.python.org/3/
Real Python: "Python Tutorials" https://realpython.com/
W3Schools: "Python Tutorial" https://www.w3schools.com/python/
Lutz, M. (2013). Learning Python. O'Reilly Media.
MongoDB Official Documentation: "MongoDB Manual" https://docs.mongodb.com/manual/
MongoDB University: "Free Courses on MongoDB" https://university.mongodb.com/
Chodorow, K. (2013). MongoDB: The Definitive Guide. O'Reilly Media.
Flask Official Documentation: "Welcome to Flask" https://flask.palletsprojects.com/
Miguel Grinberg's Flask Mega-Tutorial: https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world
Real Python: "Flask by Example" https://realpython.com/flask-by-example-part-1-project-setup/
Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.
Full Stack Python: "Full Stack Web Development with Python" https://www.fullstackpython.com/
Wargo, D. (2020). Full-Stack Web Development with Python. Packt Publishing.

----------X----------